



US009342782B2

(12) **United States Patent**  
**Wegrzyn et al.**

(10) **Patent No.:** **US 9,342,782 B2**  
(45) **Date of Patent:** **May 17, 2016**

(54) **STOCHASTIC DELAY PLASTICITY**

(56) **References Cited**

(71) Applicant: **QUALCOMM Incorporated**, San Diego, CA (US)

U.S. PATENT DOCUMENTS

(72) Inventors: **Jan Krzys Wegrzyn**, San Diego, CA (US); **Regan Blythe Towal**, San Diego, CA (US); **Benjamin Samuel Schwartz**, San Diego, CA (US)

8,103,602	B2	1/2012	Izhikevich
8,447,714	B2	5/2013	Breitwisch et al.
2013/0073491	A1	3/2013	Izhikevich et al.
2013/0073495	A1	3/2013	Izhikevich et al.
2013/0073497	A1	3/2013	Akopyan et al.

(73) Assignee: **QUALCOMM INCORPORATED**, San Diego, CA (US)

FOREIGN PATENT DOCUMENTS

(\*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 257 days.

WO	2013170036	A2	11/2013
WO	2013188788	A2	12/2013

OTHER PUBLICATIONS

(21) Appl. No.: **14/228,097**

(22) Filed: **Mar. 27, 2014**

(65) **Prior Publication Data**

US 2015/0242744 A1 Aug. 27, 2015

Wu et al., Chapter 7: Learning Mechanisms in Networks of Spiking Neurons, Studies in Computational Intelligence (SCI) 35, 171-197 (2006).\*

Eurich C W., et al. "Dynamic of Self-Organized Delay Adaptation", Physical Review Letters, vol. 82, No. 7, Feb. 1999, pp. 1594-1597.

International Search Report and Written Opinion—PCT/US2015/016655—ISA/EPO—Jun. 18, 2015.

(Continued)

**Related U.S. Application Data**

(60) Provisional application No. 61/943,165, filed on Feb. 21, 2014.

*Primary Examiner* — Wilbert L Starks

(74) *Attorney, Agent, or Firm* — Seyfarth Shaw LLP

(51) **Int. Cl.**

**G06N 3/02** (2006.01)

**G06N 3/08** (2006.01)

**G06N 3/04** (2006.01)

(52) **U.S. Cl.**

CPC ..... **G06N 3/08** (2013.01); **G06N 3/049** (2013.01); **G06N 3/0472** (2013.01); **G06N 3/02** (2013.01)

(58) **Field of Classification Search**

None

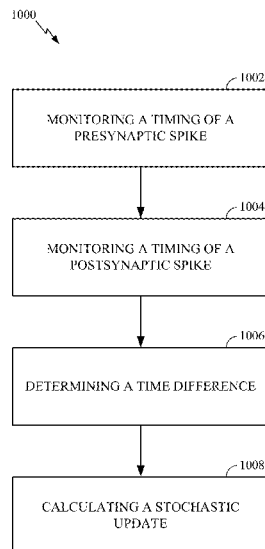
See application file for complete search history.

(57)

**ABSTRACT**

A method of operating a spiking neural network having neurons coupled together with a synapse includes monitoring a timing of a presynaptic spike and monitoring a timing of a postsynaptic spike. The method also includes determining a time difference between the postsynaptic spike and the presynaptic spike. The method further includes calculating a stochastic update of a delay for the synapse based on the time difference between the postsynaptic spike and the presynaptic spike.

**25 Claims, 10 Drawing Sheets**



(56)

**References Cited**

OTHER PUBLICATIONS

Kavehei O., et al., "Highly Scalable Neuromorphic Hardware with 1-bit Stochastic Nano-Synapses", arXiv: 1309.6419v2, Oct. 18, 2013, XP055194658, Retrieved from the Internet: URL: <http://arxiv.org/abs/1309.6419v2> [retrieved on Jun. 9, 2015].

Wright P.W., et al., "Learning Transmission Delays in Spiking Neural Networks: A Novel Approach to Sequence Learning Based on Spike Delay Variance", Proceedings of the 2012 International Joint Conference on Neural Networks (IJCNN '12), Jun. 10, 2012, XP032210066, DOI: 10.1109/IJCNN.2012.6252371.

\* cited by examiner

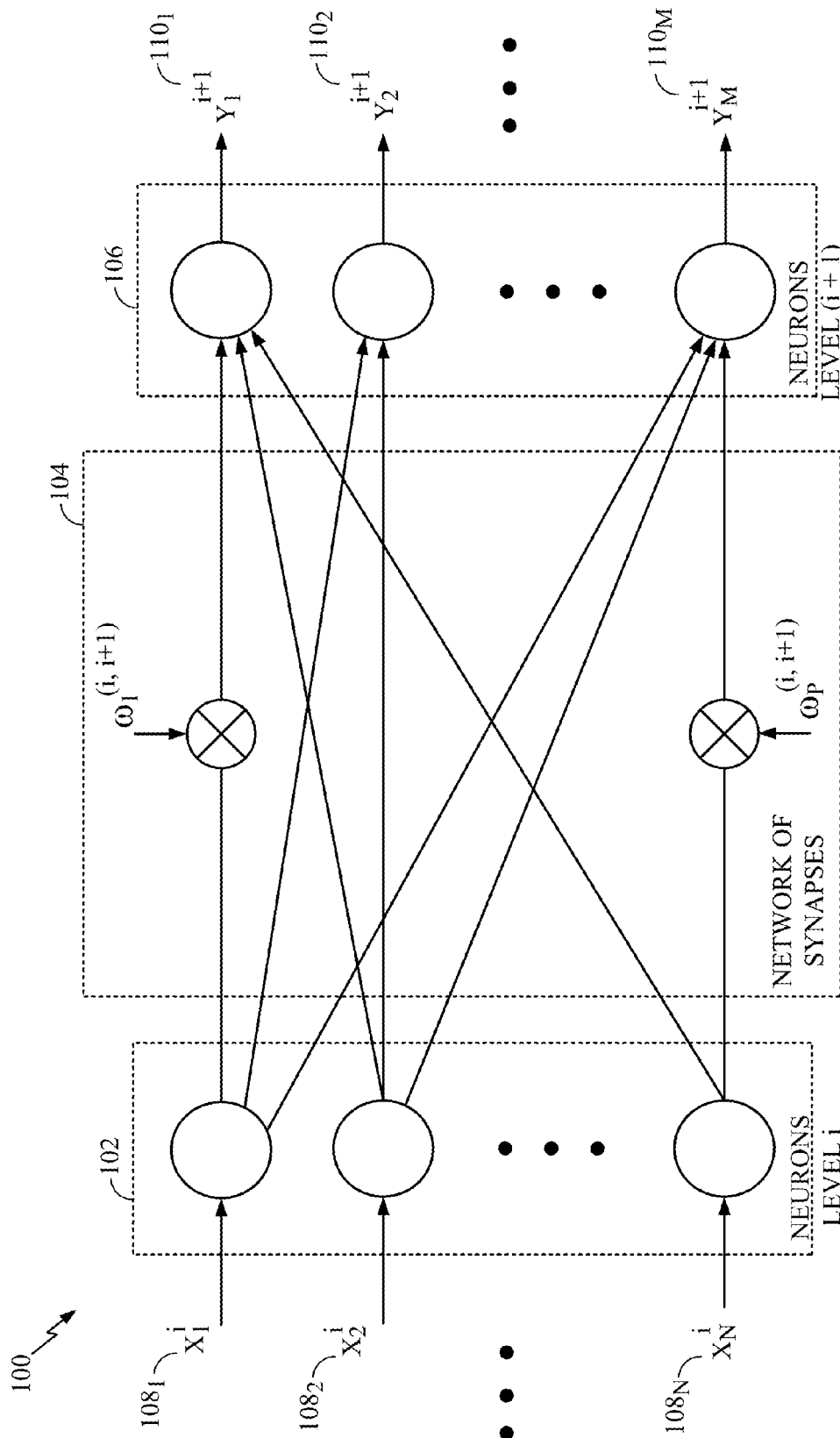
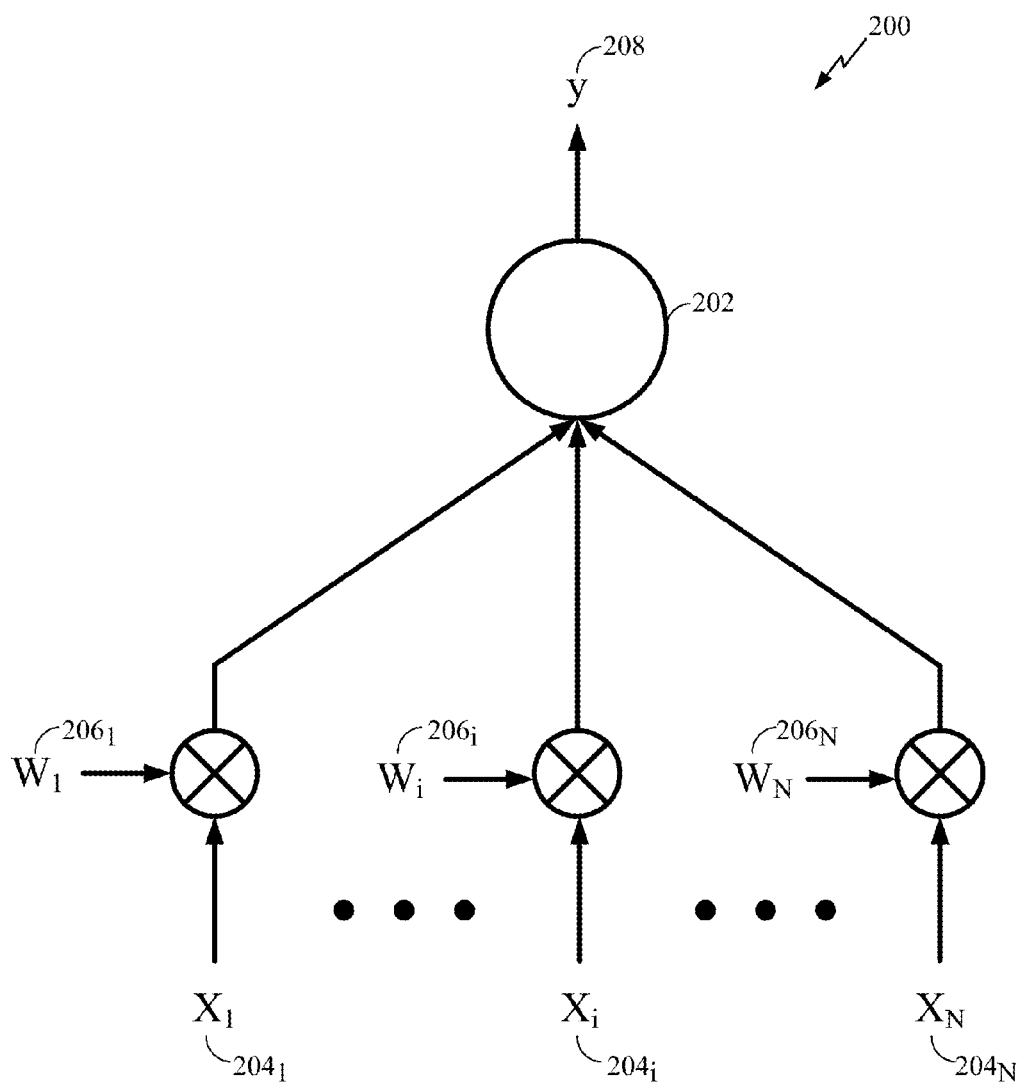
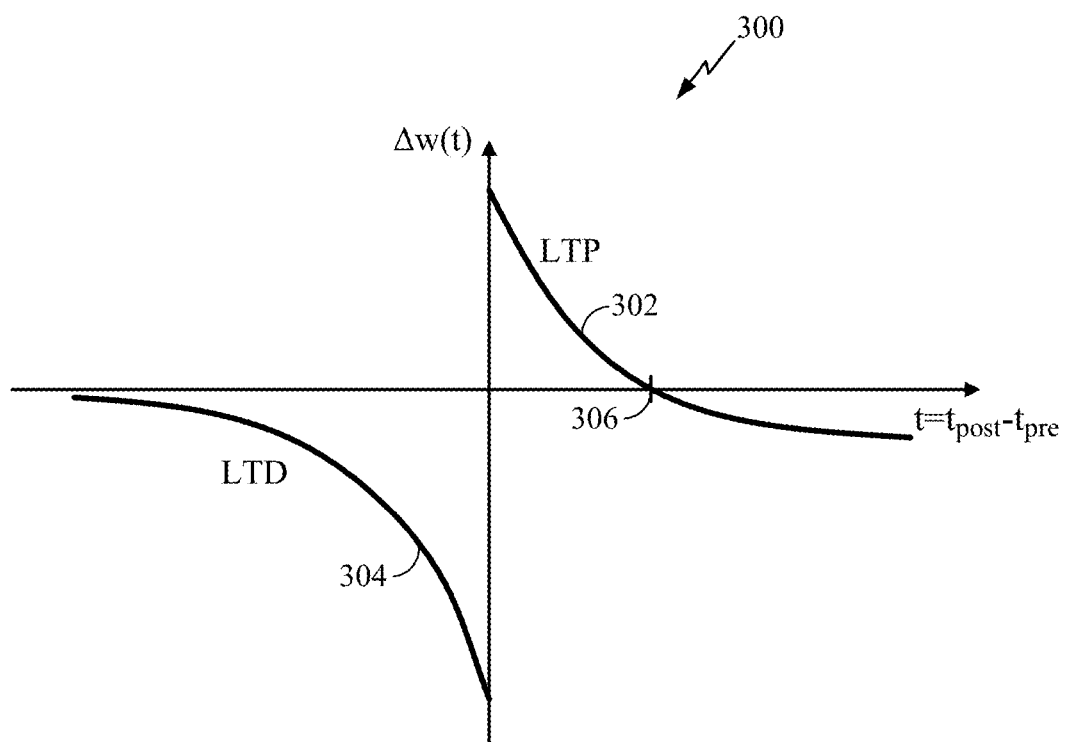
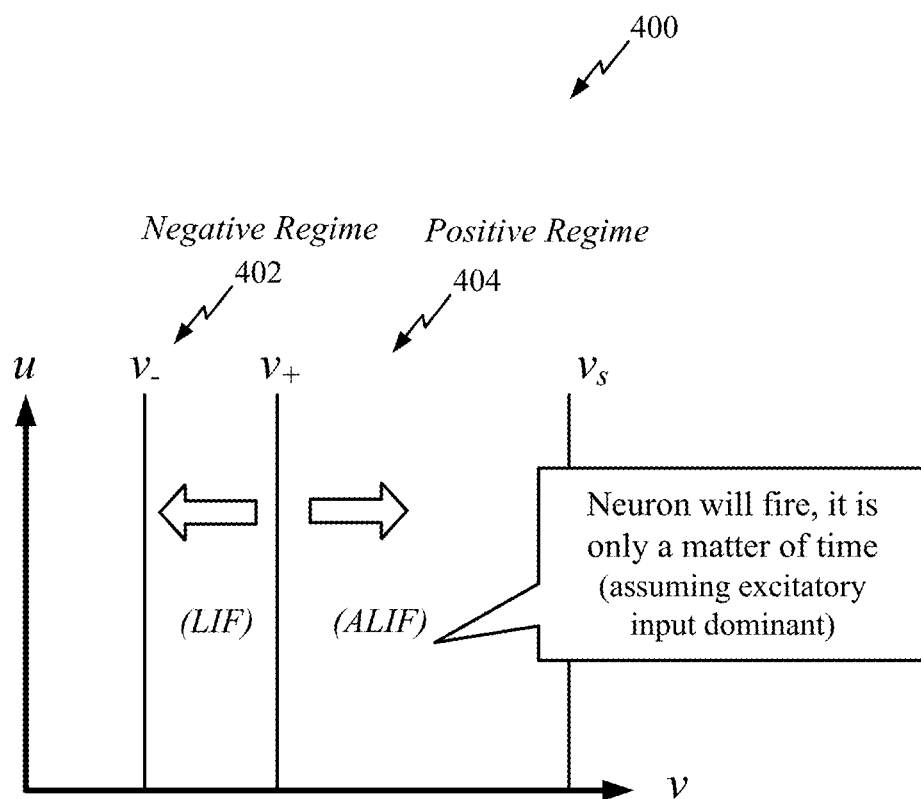


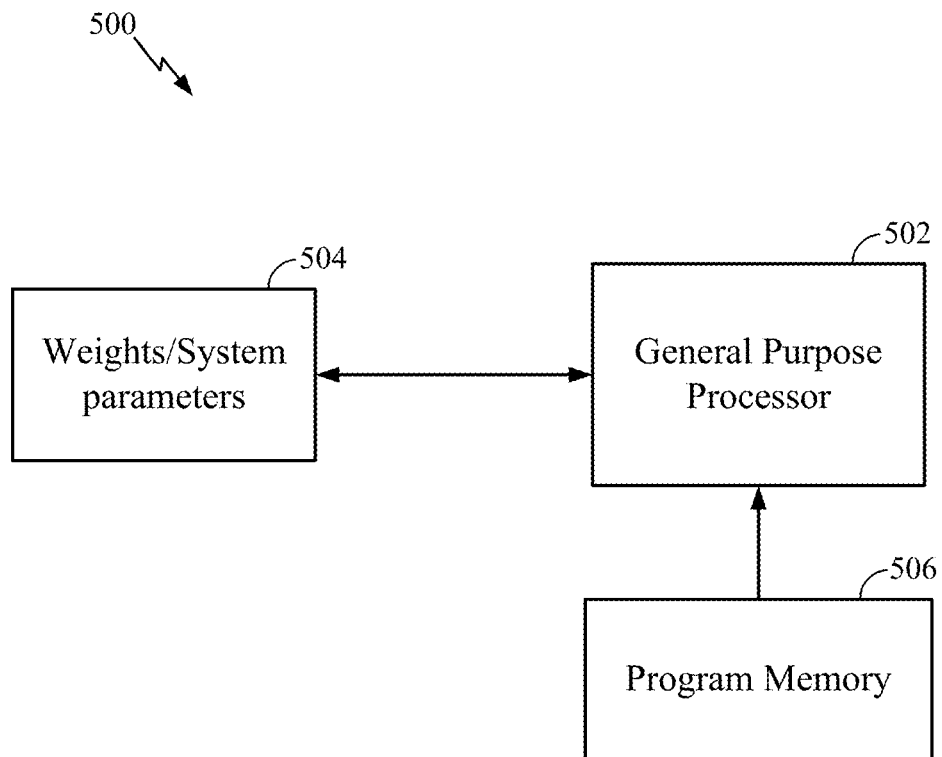
FIG. 1

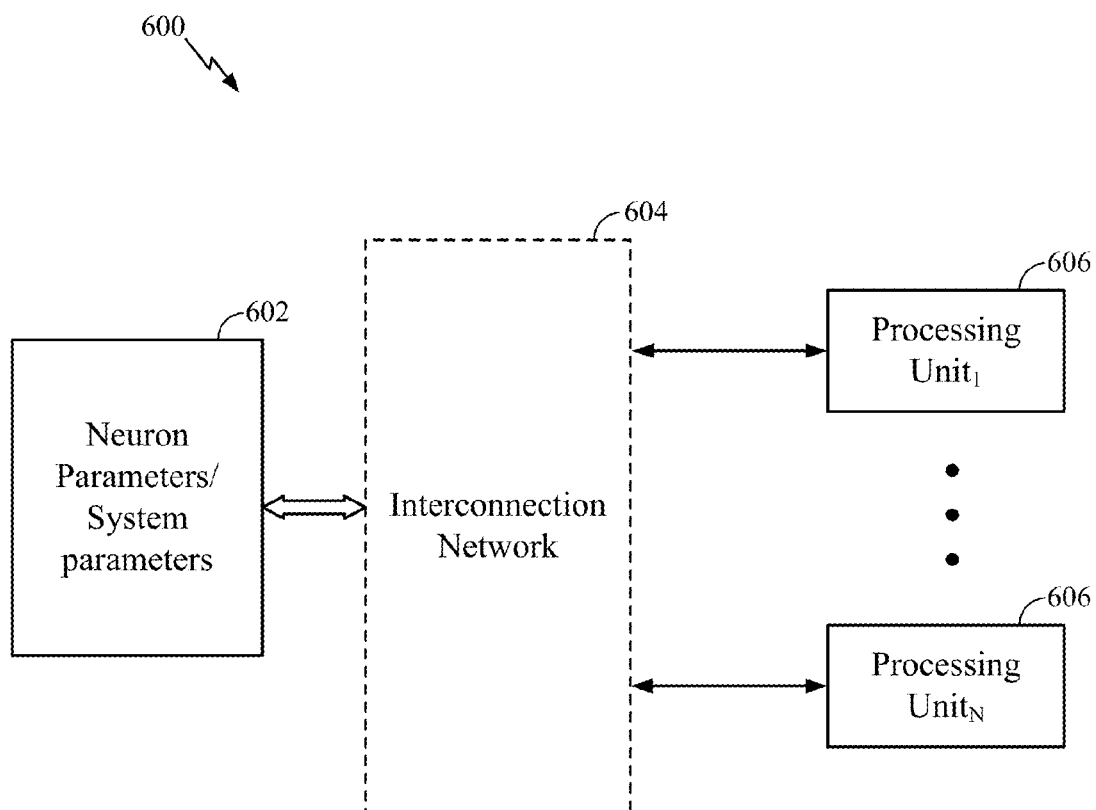
**FIG. 2**

**FIG. 3**

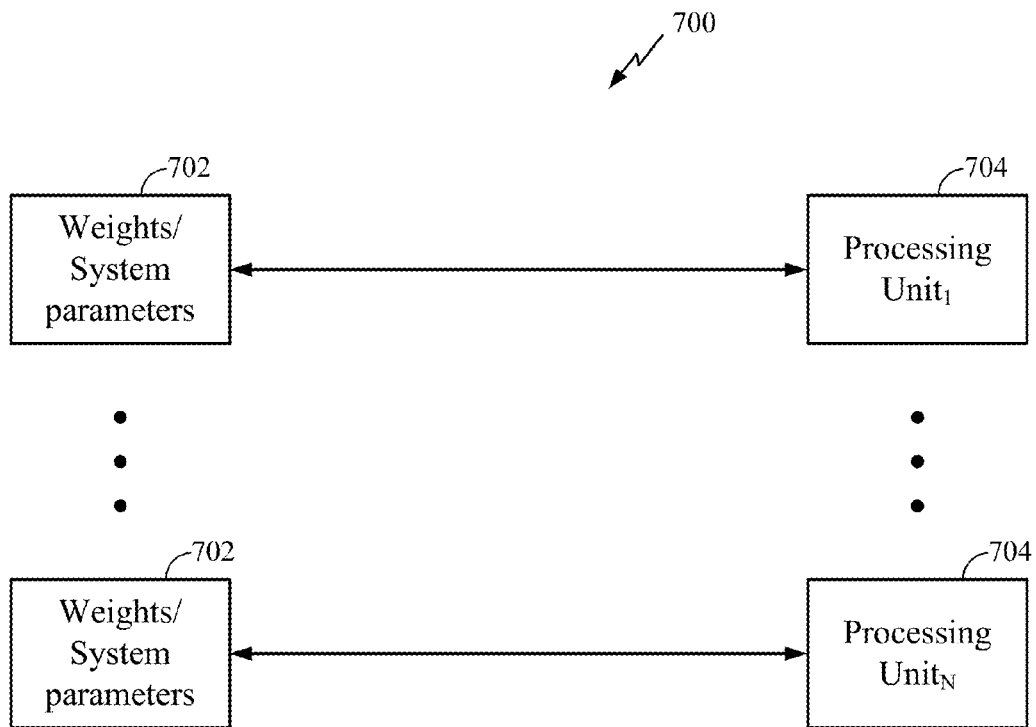


**FIG. 4**

**FIG. 5**

**FIG. 6**



**FIG. 7**

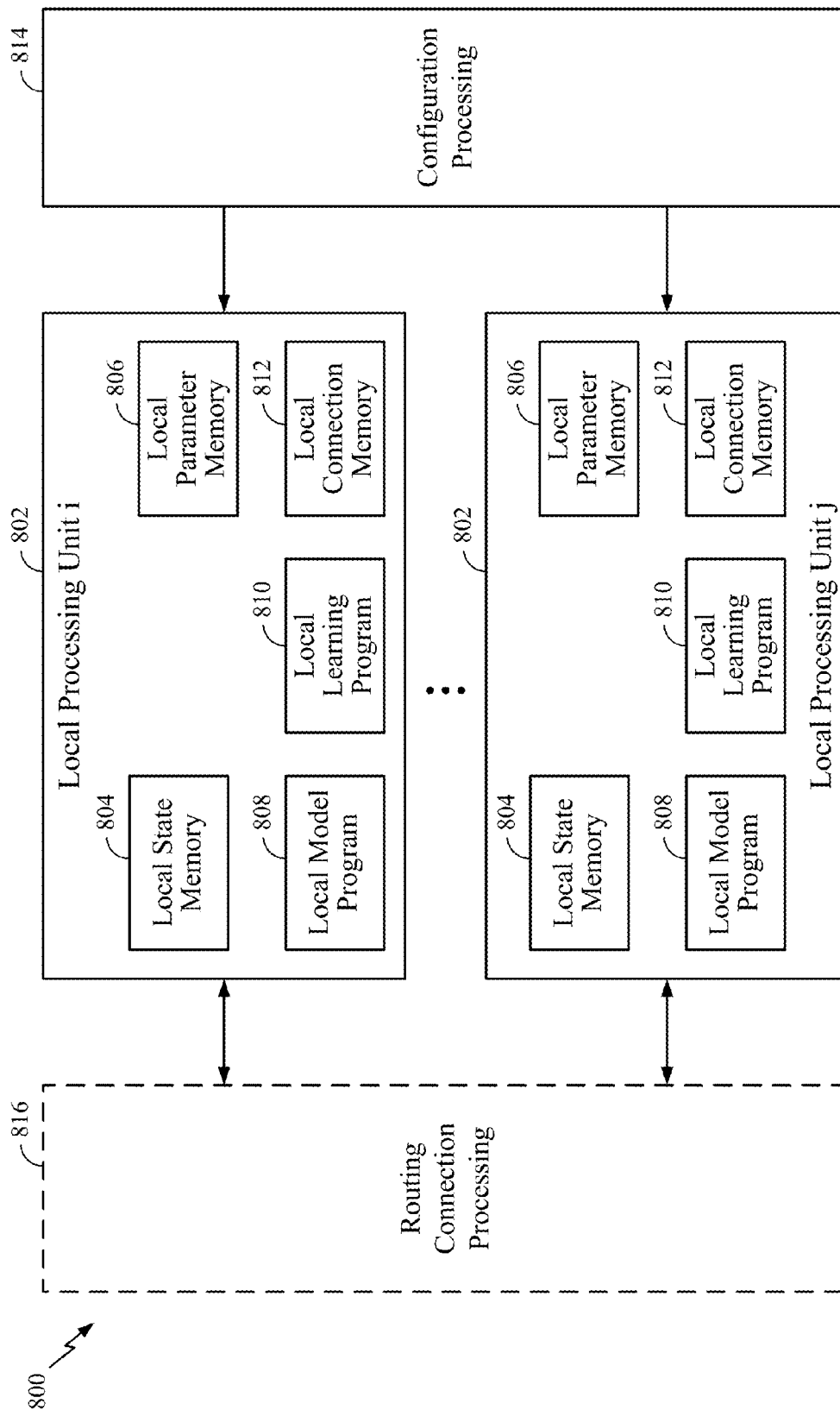


FIG. 8

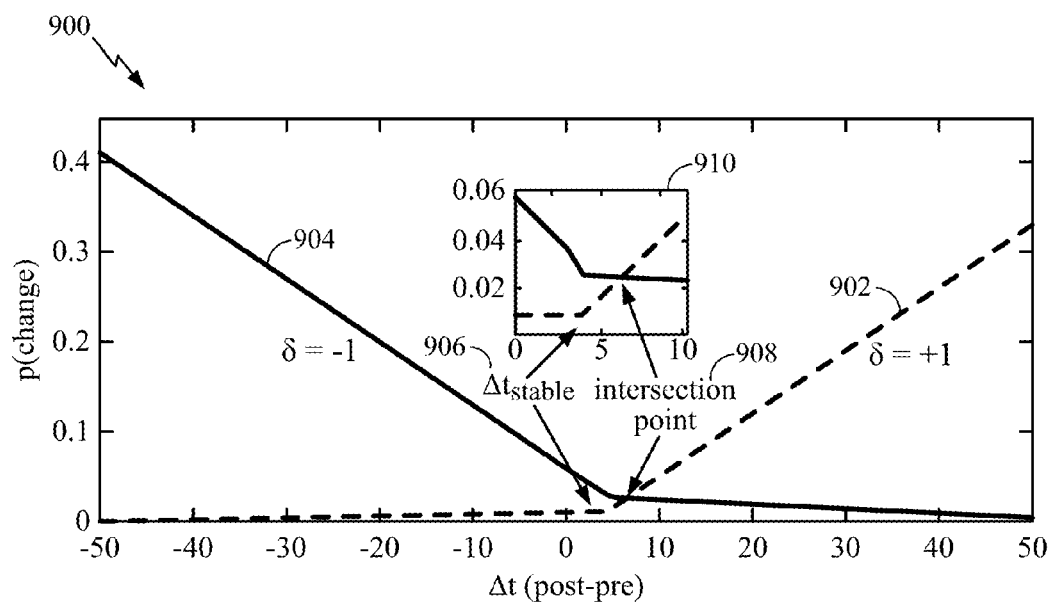


FIG. 9A

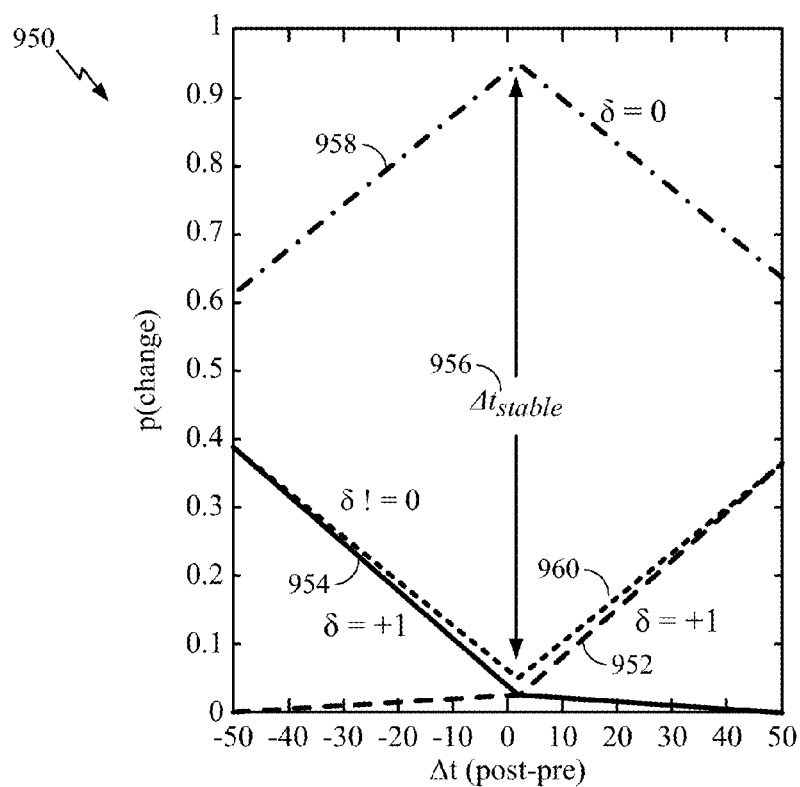
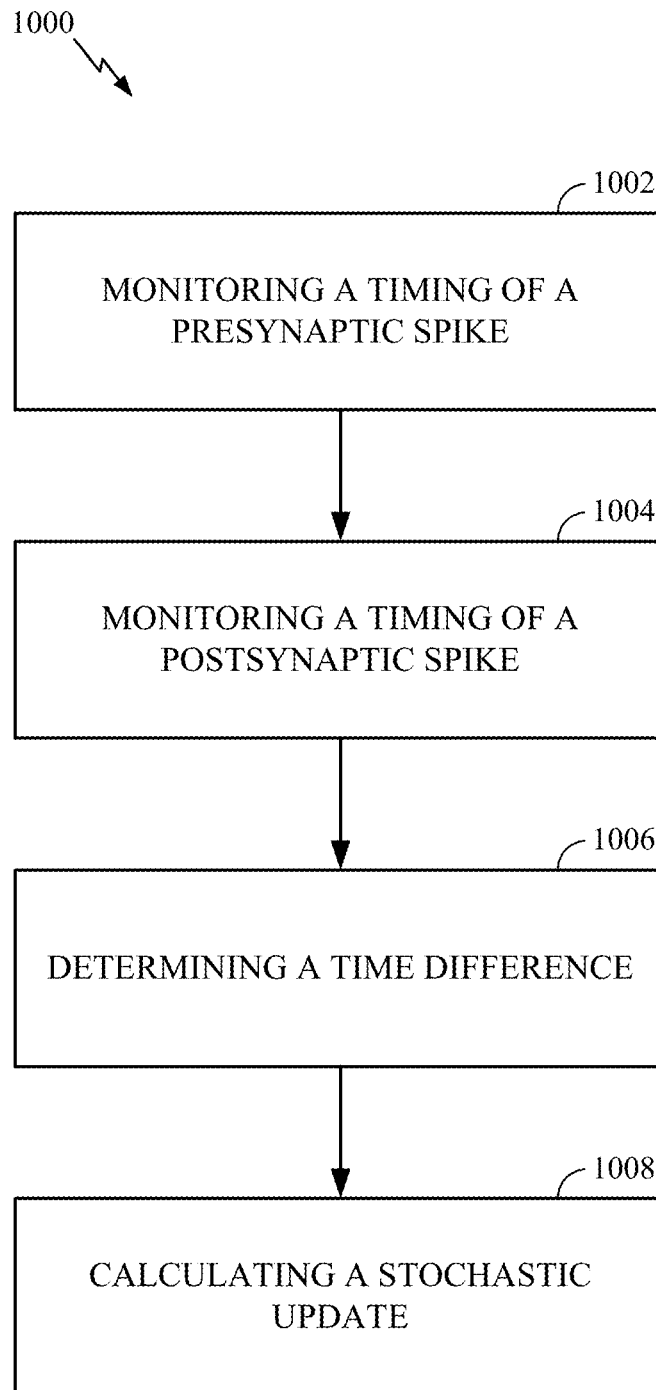


FIG. 9B

**FIG. 10**

1

**STOCHASTIC DELAY PLASTICITY****CROSS-REFERENCE TO RELATED APPLICATION**

The present application claims the benefit of U.S. Provisional Patent Application No. 61/943,165, filed on Feb. 21, 2014 and titled "STOCHASTIC DELAY PLASTICITY," the disclosure of which is expressly incorporated by reference herein in its entirety.

**BACKGROUND****1. Field**

Certain aspects of the present disclosure generally relate to neural system engineering and, more particularly, to systems and methods for designing and operating a neural network using stochastic delay plasticity.

**2. Background**

An artificial neural network, which may comprise an interconnected group of artificial neurons (i.e., neuron models), is a computational device or represents a method to be performed by a computational device. Artificial neural networks may have corresponding structure and/or function in biological neural networks. However, artificial neural networks may provide innovative and useful computational techniques for certain applications in which traditional computational techniques are cumbersome, impractical, or inadequate. Because artificial neural networks can infer a function from observations, such networks are particularly useful in applications where the complexity of the task or data makes the design of the function by conventional techniques burdensome.

Spike-timing dependent delay plasticity is a technique to alter the time at which incoming information from a presynaptic neuron arrives at or is updated in a postsynaptic neuron. In many cases, the function of delay plasticity is to cause temporally separated events to arrive at the postsynaptic neuron at the same time, which increases the likelihood that the neuron will fire an action potential. Delay plasticity is typically implemented by changing the delay of the synapse (or connection) between pairs of pre- and postsynaptic neurons. Spike-timing dependent delay plasticity is a special case of delay plasticity where the change in delay is determined by the time difference between the pre- and post-synaptic spike times. In past implementations, the delay change was deterministic; meaning that for a given time difference between the pre- and postsynaptic neurons the delay change was a pre-determined value. Implementations typically rely on a positive delay change when the pre-fires before the postsynaptic neuron and a negative delay change in the reverse case. While this approach works well when the postsynaptic neuron fires in the middle of a group of presynaptic spikes, invoking both positive and negative delay changes, when the postsynaptic spike fires after a group of presynaptic spikes, only positive delay changes are invoked. This produces ever increasing delays that can saturate or add unnecessary delay to the system (e.g., producing synaptic delays of 25, 27, and 29 when delays of 1, 3, and 5 would provide the same functionality). This gratuitous delay occurs because there is no mechanism to minimize the overall delay after the presynaptic spikes have been clustered together.

**SUMMARY**

In an aspect of the present disclosure, a method of operating a spiking neural network having neurons coupled together with a synapse is disclosed. The method includes monitoring

2

a timing of a presynaptic spike and monitoring a timing of a postsynaptic spike. The method also includes determining a time difference between the postsynaptic spike and the presynaptic spike. The method further includes calculating a stochastic update of a delay for the synapse based on the time difference.

In another aspect of the present disclosure, an apparatus for operating a spiking neural network having neurons coupled together with a synapse is disclosed. The apparatus has a memory and at least one processor coupled to the memory. The processor(s) is configured to monitor a timing of a presynaptic spike and monitor a timing of a postsynaptic spike. The processor(s) is also configured to determine a time difference between the postsynaptic spike and the presynaptic spike. The processor(s) is further configured to calculate a stochastic update of a delay for the synapse based on the time difference.

In yet another aspect of the present disclosure, an apparatus for operating a spiking neural network having neurons coupled together with a synapse is disclosed. The apparatus has means for monitoring a timing of a presynaptic spike and monitoring a timing of a postsynaptic spike. The apparatus also has means for determining a time difference between the postsynaptic spike and the presynaptic spike. The apparatus further has means for calculating a stochastic update of a delay for the synapse based on the time difference.

In still another aspect of the present disclosure, a computer program product for operating a spiking neural network having neurons coupled together with a synapse is disclosed. The computer program product includes a non-transitory computer readable medium having encoded thereon program code. The program code includes program code to monitor a timing of a presynaptic spike and program code to monitor a timing of a postsynaptic spike. The program code also includes program code to determine a time difference between the postsynaptic spike and the presynaptic spike. The program code further includes program code to calculate a stochastic update of a delay for the synapse based on the time difference.

This has outlined, rather broadly, the features and technical advantages of the present disclosure in order that the detailed description that follows may be better understood. Additional features and advantages of the disclosure will be described below. It should be appreciated by those skilled in the art that this disclosure may be readily utilized as a basis for modifying or designing other structures for carrying out the same purposes of the present disclosure. It should also be realized by those skilled in the art that such equivalent constructions do not depart from the teachings of the disclosure as set forth in the appended claims. The novel features, which are believed to be characteristic of the disclosure, both as to its organization and method of operation, together with further objects and advantages, will be better understood from the following description when considered in connection with the accompanying figures. It is to be expressly understood, however, that each of the figures is provided for the purpose of illustration and description only and is not intended as a definition of the limits of the present disclosure.

**BRIEF DESCRIPTION OF THE DRAWINGS**

The features, nature, and advantages of the present disclosure will become more apparent from the detailed description set forth below when taken in conjunction with the drawings in which like reference characters identify correspondingly throughout.

FIG. 1 illustrates an example network of neurons in accordance with certain aspects of the present disclosure.

FIG. 2 illustrates an example of a processing unit (neuron) of a computational network (neural system or neural network) in accordance with certain aspects of the present disclosure.

FIG. 3 illustrates an example of spike-timing dependent plasticity (STDP) curve in accordance with certain aspects of the present disclosure.

FIG. 4 illustrates an example of a positive regime and a negative regime for defining behavior of a neuron model in accordance with certain aspects of the present disclosure.

FIG. 5 illustrates an example implementation of designing a neural network using a general-purpose processor in accordance with certain aspects of the present disclosure.

FIG. 6 illustrates an example implementation of designing a neural network where a memory may be interfaced with individual distributed processing units in accordance with certain aspects of the present disclosure.

FIG. 7 illustrates an example implementation of designing a neural network based on distributed memories and distributed processing units in accordance with certain aspects of the present disclosure.

FIG. 8 illustrates an example implementation of a neural network in accordance with certain aspects of the present disclosure.

FIGS. 9A-9B are diagrams illustrating exemplary stochastic delay plasticity curves in accordance with aspects of the present disclosure.

FIG. 10 is a block diagram illustrating a method for operating a neural network in accordance with an aspect of the present disclosure.

#### DETAILED DESCRIPTION

The detailed description set forth below, in connection with the appended drawings, is intended as a description of various configurations and is not intended to represent the only configurations in which the concepts described herein may be practiced. The detailed description includes specific details for the purpose of providing a thorough understanding of the various concepts. However, it will be apparent to those skilled in the art that these concepts may be practiced without these specific details. In some instances, well-known structures and components are shown in block diagram form in order to avoid obscuring such concepts.

Based on the teachings, one skilled in the art should appreciate that the scope of the disclosure is intended to cover any aspect of the disclosure, whether implemented independently or of or combined with any other aspect of the disclosure. For example, an apparatus may be implemented or a method may be practiced using any number of the aspects set forth. In addition, the scope of the disclosure is intended to cover such an apparatus or method practiced using other structure, functionality, or structure and functionality in addition to or other than the various aspects of the disclosure set forth. It should be understood that any aspect of the disclosure disclosed may be embodied by one or more elements of a claim.

The word “exemplary” is used herein to mean “serving as an example, instance, or illustration.” Any aspect described herein as “exemplary” is not necessarily to be construed as preferred or advantageous over other aspects.

Although particular aspects are described herein, many variations and permutations of these aspects fall within the scope of the disclosure. Although some benefits and advantages of the preferred aspects are mentioned, the scope of the disclosure is not intended to be limited to particular benefits,

uses or objectives. Rather, aspects of the disclosure are intended to be broadly applicable to different technologies, system configurations, networks and protocols, some of which are illustrated by way of example in the figures and in the following description of the preferred aspects. The detailed description and drawings are merely illustrative of the disclosure rather than limiting, the scope of the disclosure being defined by the appended claims and equivalents thereof.

#### An Example Neural System, Training and Operation

FIG. 1 illustrates an example artificial neural system **100** with multiple levels of neurons in accordance with certain aspects of the present disclosure. The neural system **100** may have a level of neurons **102** connected to another level of neurons **106** through a network of synaptic connections **104** (i.e., feed-forward connections). For simplicity, only two levels of neurons are illustrated in FIG. 1, although fewer or more levels of neurons may exist in a neural system. It should be noted that some of the neurons may connect to other neurons of the same layer through lateral connections. Furthermore, some of the neurons may connect back to a neuron of a previous layer through feedback connections.

As illustrated in FIG. 1, each neuron in the level **102** may receive an input signal **108** that may be generated by neurons of a previous level (not shown in FIG. 1). The signal **108** may represent an input current of the level **102** neuron. This current may be accumulated on the neuron membrane to charge a membrane potential. When the membrane potential reaches its threshold value, the neuron may fire and generate an output spike to be transferred to the next level of neurons (e.g., the level **106**). In some modeling approaches, the neuron may continuously transfer a signal to the next level of neurons. This signal is typically a function of the membrane potential. Such behavior can be emulated or simulated in hardware and/or software, including analog and digital implementations such as those described below.

In biological neurons, the output spike generated when a neuron fires is referred to as an action potential. This electrical signal is a relatively rapid, transient, nerve impulse, having an amplitude of roughly 100 mV and a duration of about 1 ms. In a particular embodiment of a neural system having a series of connected neurons (e.g., the transfer of spikes from one level of neurons to another in FIG. 1), every action potential has basically the same amplitude and duration, and thus, the information in the signal may be represented only by the frequency and number of spikes, or the time of spikes, rather than by the amplitude. The information carried by an action potential may be determined by the spike, the neuron that spiked, and the time of the spike relative to other spike or spikes. The importance of the spike may be determined by a weight applied to a connection between neurons, as explained below.

The transfer of spikes from one level of neurons to another may be achieved through the network of synaptic connections (or simply “synapses”) **104**, as illustrated in FIG. 1. Relative to the synapses **104**, neurons of level **102** may be considered presynaptic neurons and neurons of level **106** may be considered postsynaptic neurons. The synapses **104** may receive output signals (i.e., spikes) from the level **102** neurons and scale those signals according to adjustable synaptic weights  $w_{1(i,i+1)}, \dots, w_{P(i,i+1)}$  where  $P$  is a total number of synaptic connections between the neurons of levels **102** and **106** and  $i$  is an indicator of the neuron level. In the example of FIG. 1,  $i$  represents neuron level **102** and  $i+1$  represents neuron level **106**. Further, the scaled signals may be combined as an input signal of each neuron in the level **106**. Every neuron in the level **106** may generate output spikes **110** based on the cor-

5

responding combined input signal. The output spikes **110** may be transferred to another level of neurons using another network of synaptic connections (not shown in FIG. 1).

Biological synapses can mediate either excitatory or inhibitory (hyperpolarizing) actions in postsynaptic neurons and can also serve to amplify neuronal signals. Excitatory signals depolarize the membrane potential (i.e., increase the membrane potential with respect to the resting potential). If enough excitatory signals are received within a certain time period to depolarize the membrane potential above a threshold, an action potential occurs in the postsynaptic neuron. In contrast, inhibitory signals generally hyperpolarize (i.e., lower) the membrane potential. Inhibitory signals, if strong enough, can counteract the sum of excitatory signals and prevent the membrane potential from reaching a threshold. In addition to counteracting synaptic excitation, synaptic inhibition can exert powerful control over spontaneously active neurons. A spontaneously active neuron refers to a neuron that spikes without further input, for example due to its dynamics or a feedback. By suppressing the spontaneous generation of action potentials in these neurons, synaptic inhibition can shape the pattern of firing in a neuron, which is generally referred to as sculpturing. The various synapses **104** may act as any combination of excitatory or inhibitory synapses, depending on the behavior desired.

The neural system **100** may be emulated by a general purpose processor, a digital signal processor (DSP), an application specific integrated circuit (ASIC), a field programmable gate array (FPGA) or other programmable logic device (PLD), discrete gate or transistor logic, discrete hardware components, a software module executed by a processor, or any combination thereof. The neural system **100** may be utilized in a large range of applications, such as image and pattern recognition, machine learning, motor control, and alike. Each neuron in the neural system **100** may be implemented as a neuron circuit. The neuron membrane charged to the threshold value initiating the output spike may be implemented, for example, as a capacitor that integrates an electrical current flowing through it.

In an aspect, the capacitor may be eliminated as the electrical current integrating device of the neuron circuit, and a smaller memristor element may be used in its place. This approach may be applied in neuron circuits, as well as in various other applications where bulky capacitors are utilized as electrical current integrators. In addition, each of the synapses **104** may be implemented based on a memristor element, where synaptic weight changes may relate to changes of the memristor resistance. With nanometer feature-sized memristors, the area of a neuron circuit and synapses may be substantially reduced, which may make implementation of a large-scale neural system hardware implementation more practical.

Functionality of a neural processor that emulates the neural system **100** may depend on weights of synaptic connections, which may control strengths of connections between neurons. The synaptic weights may be stored in a non-volatile memory in order to preserve functionality of the processor after being powered down. In an aspect, the synaptic weight memory may be implemented on a separate external chip from the main neural processor chip. The synaptic weight memory may be packaged separately from the neural processor chip as a replaceable memory card. This may provide diverse functionalities to the neural processor, where a particular functionality may be based on synaptic weights stored in a memory card currently attached to the neural processor.

FIG. 2 illustrates an exemplary diagram **200** of a processing unit (e.g., a neuron or neuron circuit) **202** of a computa-

6

tional network (e.g., a neural system or a neural network) in accordance with certain aspects of the present disclosure. For example, the neuron **202** may correspond to any of the neurons of levels **102** and **106** from FIG. 1. The neuron **202** may receive multiple input signals **204**<sub>1</sub>-**204**<sub>N</sub>, which may be signals external to the neural system, or signals generated by other neurons of the same neural system, or both. The input signal may be a current, a conductance, a voltage, a real-valued, and/or a complex-valued. The input signal may comprise a numerical value with a fixed-point or a floating-point representation. These input signals may be delivered to the neuron **202** through synaptic connections that scale the signals according to adjustable synaptic weights **206**<sub>1</sub>-**206**<sub>N</sub> ( $W_1$ - $W_N$ ), where N may be a total number of input connections of the neuron **202**.

The neuron **202** may combine the scaled input signals and use the combined scaled inputs to generate an output signal **208** (i.e., a signal Y). The output signal **208** may be a current, a conductance, a voltage, a real-valued and/or a complex-valued. The output signal may be a numerical value with a fixed-point or a floating-point representation. The output signal **208** may be then transferred as an input signal to other neurons of the same neural system, or as an input signal to the same neuron **202**, or as an output of the neural system.

The processing unit (neuron) **202** may be emulated by an electrical circuit, and its input and output connections may be emulated by electrical connections with synaptic circuits. The processing unit **202** and its input and output connections may also be emulated by a software code. The processing unit **202** may also be emulated by an electric circuit, whereas its input and output connections may be emulated by a software code. In an aspect, the processing unit **202** in the computational network may be an analog electrical circuit. In another aspect, the processing unit **202** may be a digital electrical circuit. In yet another aspect, the processing unit **202** may be a mixed-signal electrical circuit with both analog and digital components. The computational network may include processing units in any of the aforementioned forms. The computational network (neural system or neural network) using such processing units may be utilized in a large range of applications, such as image and pattern recognition, machine learning, motor control, and the like.

During the course of training a neural network, synaptic weights (e.g., the weights  $w_1^{(i,j+1)}, \dots, w_p^{(i,j+1)}$  from FIG. 1 and/or the weights **206**<sub>1</sub>-**206**<sub>N</sub> from FIG. 2) may be initialized with random values and increased or decreased according to a learning rule. Those skilled in the art will appreciate that examples of the learning rule include, but are not limited to the spike-timing-dependent plasticity (STDP) learning rule, the Hebb rule, the Oja rule, the Bienenstock-Copper-Munro (BCM) rule, etc. In certain aspects, the weights may settle or converge to one of two values (i.e., a bimodal distribution of weights). This effect can be utilized to reduce the number of bits for each synaptic weight, increase the speed of reading and writing from/to a memory storing the synaptic weights, and to reduce power and/or processor consumption of the synaptic memory.

Synapse Type

In hardware and software models of neural networks, the processing of synapse related functions can be based on synaptic type. Synapse types may be non-plastic synapses (no changes of weight and delay), plastic synapses (weight may change), structural delay plastic synapses (weight and delay may change), fully plastic synapses (weight, delay and connectivity may change), and variations thereupon (e.g., delay may change, but no change in weight or connectivity). The advantage of multiple types is that processing can be subdi-

vided. For example, non-plastic synapses may not utilize plasticity functions to be executed (or waiting for such functions to complete). Similarly, delay and weight plasticity may be subdivided into operations that may operate together or separately, in sequence or in parallel. Different types of synapses may have different lookup tables or formulas and parameters for each of the different plasticity types that apply. Thus, the methods would access the relevant tables, formulas, or parameters for the synapse's type.

There are further implications of the fact that spike-timing dependent structural plasticity may be executed independently of synaptic plasticity. Structural plasticity may be executed even if there is no change to weight magnitude (e.g., if the weight has reached a minimum or maximum value, or it is not changed due to some other reason) s structural plasticity (i.e., an amount of delay change) may be a direct function of pre-post spike time difference. Alternatively, structural plasticity may be set as a function of the weight change amount or based on conditions relating to bounds of the weights or weight changes. For example, a synapse delay may change only when a weight change occurs or if weights reach zero but not if they are at a maximum value. However, it may be advantageous to have independent functions so that these processes can be parallelized reducing the number and overlap of memory accesses.

#### Determination of Synaptic Plasticity

Neuroplasticity (or simply "plasticity") is the capacity of neurons and neural networks in the brain to change their synaptic connections and behavior in response to new information, sensory stimulation, development, damage, or dysfunction. Plasticity is important to learning and memory in biology, as well as for computational neuroscience and neural networks. Various forms of plasticity have been studied, such as synaptic plasticity (e.g., according to the Hebbian theory), spike-timing-dependent plasticity (STDP), non-synaptic plasticity, activity-dependent plasticity, structural plasticity and homeostatic plasticity.

STDP is a learning process that adjusts the strength of synaptic connections between neurons. The connection strengths are adjusted based on the relative timing of a particular neuron's output and received input spikes (i.e., action potentials). Under the STDP process, long-term potentiation (LTP) may occur if an input spike to a certain neuron tends, on average, to occur immediately before that neuron's output spike. Then, that particular input is made somewhat stronger. On the other hand, long-term depression (LTD) may occur if an input spike tends, on average, to occur immediately after an output spike. Then, that particular input is made somewhat weaker, and hence the name "spike-timing-dependent plasticity." Consequently, inputs that might be the cause of the postsynaptic neuron's excitation are made even more likely to contribute in the future, whereas inputs that are not the cause of the postsynaptic spike are made less likely to contribute in the future. The process continues until a subset of the initial set of connections remains, while the influence of all others is reduced to an insignificant level.

Because a neuron generally produces an output spike when many of its inputs occur within a brief period (i.e., being cumulative sufficient to cause the output), the subset of inputs that typically remains includes those that tended to be correlated in time. In addition, because the inputs that occur before the output spike are strengthened, the inputs that provide the earliest sufficiently cumulative indication of correlation will eventually become the final input to the neuron.

The STDP learning rule may effectively adapt a synaptic weight of a synapse connecting a presynaptic neuron to a postsynaptic neuron as a function of time difference between

spike time  $t_{pre}$  of the presynaptic neuron and spike time  $t_{post}$  of the postsynaptic neuron (i.e.,  $t = t_{post} - t_{pre}$ ). A typical formulation of the STDP is to increase the synaptic weight (i.e., potentiate the synapse) if the time difference is positive (the presynaptic neuron fires before the postsynaptic neuron), and decrease the synaptic weight (i.e., depress the synapse) if the time difference is negative (the postsynaptic neuron fires before the presynaptic neuron).

In the STDP process, a change of the synaptic weight over time may be typically achieved using an exponential decay, as given by:

$$\Delta w(t) = \begin{cases} a_+ e^{-t/k_+} + \mu, & t > 0 \\ a_- e^{t/k_-}, & t < 0 \end{cases}, \quad (1)$$

where  $k_+$  and  $k_- \tau_{sign(\Delta t)}$  are time constants for positive and negative time difference, respectively,  $a_+$  and  $a_-$  are corresponding scaling magnitudes, and  $\mu$  is an offset that may be applied to the positive time difference and/or the negative time difference.

FIG. 3 illustrates an exemplary diagram 300 of a synaptic weight change as a function of relative timing of presynaptic and postsynaptic spikes in accordance with the STDP. If a presynaptic neuron fires before a postsynaptic neuron, then a corresponding synaptic weight may be increased, as illustrated in a portion 302 of the graph 300. This weight increase can be referred to as an LTP of the synapse. It can be observed from the graph portion 302 that the amount of LTP may decrease roughly exponentially as a function of the difference between presynaptic and postsynaptic spike times. The reverse order of firing may reduce the synaptic weight, as illustrated in a portion 304 of the graph 300, causing an LTD of the synapse.

As illustrated in the graph 300 in FIG. 3, a negative offset  $\mu$  may be applied to the LTP (causal) portion 302 of the STDP graph. A point of cross-over 306 of the x-axis ( $y=0$ ) may be configured to coincide with the maximum time lag for considering correlation for causal inputs from layer  $i-1$ . In the case of a frame-based input (i.e., an input that is in the form of a frame of a particular duration comprising spikes or pulses), the offset value  $\mu$  can be computed to reflect the frame boundary. A first input spike (pulse) in the frame may be considered to decay over time either as modeled by a postsynaptic potential directly or in terms of the effect on neural state. If a second input spike (pulse) in the frame is considered correlated or relevant to a particular time frame, then the relevant times before and after the frame may be separated at that time frame boundary and treated differently in plasticity terms by offsetting one or more parts of the STDP curve such that the value in the relevant times may be different (e.g., negative for greater than one frame and positive for less than one frame). For example, the negative offset  $\mu$  may be set to offset LTP such that the curve actually goes below zero at a pre-post time greater than the frame time and it is thus part of LTD instead of LTP.

#### Neuron Models and Operation

There are some general principles for designing a useful spiking neuron model. A good neuron model may have rich potential behavior in terms of two computational regimes: coincidence detection and functional computation. Moreover, a good neuron model should have two elements to allow temporal coding: arrival time of inputs affects output time and coincidence detection can have a narrow time window. Finally, to be computationally attractive, a good neuron



model may have a closed-form solution in continuous time and stable behavior including near attractors and saddle points. In other words, a useful neuron model is one that is practical and that can be used to model rich, realistic and biologically-consistent behaviors, as well as be used to both engineer and reverse engineer neural circuits.

A neuron model may depend on events, such as an input arrival, output spike or other event whether internal or external. To achieve a rich behavioral repertoire, a state machine that can exhibit complex behaviors may be desired. If the occurrence of an event itself, separate from the input contribution (if any), can influence the state machine and constrain dynamics subsequent to the event, then the future state of the system is not only a function of a state and input, but rather a function of a state, event, and input.

In an aspect, a neuron  $n$  may be modeled as a spiking leaky-integrate-and-fire neuron with a membrane voltage  $v_n(t)$  governed by the following dynamics:

$$\frac{dv_n(t)}{dt} = \alpha v_n(t) + \beta \sum_m w_{m,n} y_m(t - \Delta t_{m,n}), \quad (2)$$

where  $\alpha$  and  $\beta$  are parameters,  $w_{m,n}$  is a synaptic weight for the synapse connecting a presynaptic neuron  $m$  to a postsynaptic neuron  $n$ , and  $y_m(t)$  is the spiking output of the neuron  $m$  that may be delayed by dendritic or axonal delay according to  $\Delta t_{m,n}$  until arrival at the neuron  $n$ 's soma.

It should be noted that there is a delay from the time when sufficient input to a postsynaptic neuron is established until the time when the postsynaptic neuron actually fires. In a dynamic spiking neuron model, such as Izhikevich's simple model, a time delay may be incurred if there is a difference between a depolarization threshold  $v_r$  and a peak spike voltage  $v_{peak}$ . For example, in the simple model, neuron soma dynamics can be governed by the pair of differential equations for voltage and recovery, i.e.:

$$\frac{dv}{dt} = (k(v - v_r)(v - v_r) - u + I) / C, \quad (3)$$

$$\frac{du}{dt} = a(b(v - v_r) - u). \quad (4)$$

where  $v$  is a membrane potential,  $u$  is a membrane recovery variable,  $k$  is a parameter that describes time scale of the membrane potential  $v$ ,  $a$  is a parameter that describes time scale of the recovery variable  $u$ ,  $b$  is a parameter that describes sensitivity of the recovery variable  $u$  to the sub-threshold fluctuations of the membrane potential  $v$ ,  $v_r$  is a membrane resting potential,  $I$  is a synaptic current, and  $C$  is a membrane's capacitance. In accordance with this model, the neuron is defined to spike when  $v > v_{peak}$ .  
Hunzinger Cold Model

The Hunzinger Cold neuron model is a minimal dual-regime spiking linear dynamical model that can reproduce a rich variety of neural behaviors. The model's one- or two-dimensional linear dynamics can have two regimes, wherein the time constant (and coupling) can depend on the regime. In the sub-threshold regime, the time constant, negative by convention, represents leaky channel dynamics generally acting to return a cell to rest in a biologically-consistent linear fashion. The time constant in the supra-threshold regime, positive

by convention, reflects anti-leaky channel dynamics generally driving a cell to spike while incurring latency in spike-generation.

As illustrated in FIG. 4, the dynamics of the model **400** may be divided into two (or more) regimes. These regimes may be called the negative regime **402** (also interchangeably referred to as the leaky-integrate-and-fire (LIF) regime, not to be confused with the LIF neuron model) and the positive regime **404** (also interchangeably referred to as the anti-leaky-integrate-and-fire (ALIF) regime, not to be confused with the ALIF neuron model). In the negative regime **402**, the state tends toward rest ( $v_-$ ) at the time of a future event. In this negative regime, the model generally exhibits temporal input detection properties and other sub-threshold behavior. In the positive regime **404**, the state tends toward a spiking event ( $v_+$ ). In this positive regime, the model exhibits computational properties, such as incurring a latency to spike depending on subsequent input events. Formulation of dynamics in terms of events and separation of the dynamics into these two regimes are fundamental characteristics of the model.

Linear dual-regime bi-dimensional dynamics (for states  $v$  and  $u$ ) may be defined by convention as:

$$\tau_\rho \frac{dv}{dt} = v + q_\rho \quad (5)$$

$$-\tau_u \frac{du}{dt} = u + r \quad (6)$$

where  $q_\rho$  and  $r$  are the linear transformation variables for coupling.

The symbol  $\rho$  is used herein to denote the dynamics regime with the convention to replace the symbol  $\rho$  with the sign “-” or “+” for the negative and positive regimes, respectively, when discussing or expressing a relation for a specific regime.

The model state is defined by a membrane potential (voltage)  $v$  and recovery current  $u$ . In basic form, the regime is essentially determined by the model state. There are subtle, but important aspects of the precise and general definition, but for the moment, consider the model to be in the positive regime **404** if the voltage  $v$  is above a threshold ( $v_+$ ) and otherwise in the negative regime **402**.

The regime-dependent time constants include  $\tau_-$  which is the negative regime time constant, and  $\tau_+$  which is the positive regime time constant. The recovery current time constant  $\tau_u$  is typically independent of regime. For convenience, the negative regime time constant  $\tau_-$  is typically specified as a negative quantity to reflect decay so that the same expression for voltage evolution may be used as for the positive regime in which the exponent and  $\tau_+$  will generally be positive, as will be  $\tau_u$ .

The dynamics of the two state elements may be coupled at events by transformations offsetting the states from their null-clines, where the transformation variables are:

$$q_\rho = -\tau_\rho \beta u - v_\rho \quad (7)$$

$$r = \delta(v + \epsilon) \quad (8)$$

where  $\delta$ ,  $\epsilon$ ,  $\beta$  and  $v_-$ ,  $v_+$  are parameters. The two values for  $v_\rho$  are the base for reference voltages for the two regimes. The parameter  $v_-$  is the base voltage for the negative regime, and the membrane potential will generally decay toward  $v_-$  in the negative regime. The parameter  $v_+$  is the base voltage for the positive regime, and the membrane potential will generally tend away from  $v_+$  in the positive regime.

The null-clines for  $v$  and  $u$  are given by the negative of the transformation variables  $q_\rho$  and  $r$ , respectively. The parameter

11

$\delta$  is a scale factor controlling the slope of the  $u$  null-cline. The parameter  $s$  is typically set equal to  $-v_-$ . The parameter  $\beta$  is a resistance value controlling the slope of the  $v$  null-clines in both regimes. The  $\tau_p$  time-constant parameters control not only the exponential decays, but also the null-cline slopes in each regime separately.

The model may be defined to spike when the voltage  $v$  reaches a value  $v_s$ . Subsequently, the state may be reset at a reset event (which may be one and the same as the spike event):

$$v = \hat{v}_- \quad (9)$$

$$u = u + \Delta u \quad (10)$$

where  $\hat{v}_-$  and  $\Delta u$  are parameters. The reset voltage  $\hat{v}_-$  is typically set to  $v_-$ .

By a principle of momentary coupling, a closed form solution is possible not only for state (and with a single exponential term), but also for the time to reach a particular state. The close form state solutions are:

$$v(t + \Delta t) = (v(t) + q_p) e^{\frac{\Delta t}{\tau_p}} - q_p \quad (11)$$

$$u(t + \Delta t) = (u(t) + r) e^{\frac{\Delta t}{\tau_u}} - r \quad (12)$$

Therefore, the model state may be updated only upon events, such as an input (presynaptic spike) or output (postsynaptic spike). Operations may also be performed at any particular time (whether or not there is input or output).

Moreover, by the momentary coupling principle, the time of a postsynaptic spike may be anticipated so the time to reach a particular state may be determined in advance without iterative techniques or Numerical Methods (e.g., the Euler numerical method). Given a prior voltage state  $v_0$ , the time delay until voltage state  $v_f$  is reached is given by:

$$\Delta t = \tau_p \log \frac{v_f + q_p}{v_0 + q_p} \quad (13)$$

If a spike is defined as occurring at the time the voltage state  $v$  reaches  $v_s$ , then the closed-form solution for the amount of time, or relative delay, until a spike occurs as measured from the time that the voltage is at a given state  $v$  is:

$$\Delta t_s = \begin{cases} \tau_+ \log \frac{v_s + q_+}{v + q_+} & \text{if } v > \hat{v}_+ \\ \infty & \text{otherwise} \end{cases} \quad (14)$$

where  $\hat{v}_+$  is typically set to parameter  $v_+$ , although other variations may be possible.

The above definitions of the model dynamics depend on whether the model is in the positive or negative regime. As mentioned, the coupling and the regime  $\rho$  may be computed upon events. For purposes of state propagation, the regime and coupling (transformation) variables may be defined based on the state at the time of the last (prior) event. For purposes of subsequently anticipating spike output time, the regime and coupling variable may be defined based on the state at the time of the next (current) event.

There are several possible implementations of the Cold model, and executing the simulation, emulation or model in time. This includes, for example, event-update, step-event

12

update, and step-update modes. An event update is an update where states are updated based on events or "event update" (at particular moments). A step update is an update when the model is updated at intervals (e.g., 1 ms). This does not necessarily utilize iterative methods or Numerical methods. An event-based implementation is also possible at a limited time resolution in a step-based simulator by only updating the model if an event occurs at or between steps or by "step-event" update.

#### Stochastic Delay Plasticity

Aspects of the present disclosure are directed to operating a neural network using stochastic delay plasticity (SDP). Delay plasticity allows the synaptic delay between two neurons to be modified or trained in order to achieve a specific functionality. One function that delay plasticity performs is to take a group of spikes occurring at different times and delay the earlier spikes such that all of the postsynaptic potentials arrive at the postsynaptic cell at the same time. In essence, delay plasticity can align a temporally diverse spike pattern.

In delay plasticity, a curve may specify an amount the delay should be changed as a function of a time difference between a presynaptic and postsynaptic spike. While this approach may work when the postsynaptic neuron fires in the middle of a group of presynaptic spikes, invoking both positive and negative delay changes, when the postsynaptic spike fires after a group of presynaptic spikes, only positive delay changes are invoked. Thus, this approach may produce ever increasing delays that can saturate or add unnecessary delay to the system (e.g., producing synaptic delays of 25, 27, and 29 when delays of 1, 3, and 5 would provide the same functionality). This gratuitous delay may occur because there is no mechanism to reduce or minimize the overall delay after the presynaptic spikes have been clustered together.

In accordance with aspects of the present disclosure, a fixed delay change may be used in which the sign of the delay change is probabilistically altered based on the time difference between the presynaptic and postsynaptic spikes. To implement stochastic delay plasticity, a curve defining a non-zero probability of choosing a positive (+) change or a negative (-) change for every post-pre spike time difference may be constructed. For example, if the postsynaptic cell spikes 5 ms after the presynaptic cell, there could be a 0.3 probability of increasing, a 0.1 probability of decreasing and a 0.6 probability of not changing the synaptic delay by a value of one. One difference between stochastic delay plasticity and other implementations of delay plasticity is that for the same input conditions ( $\Delta t = \Delta t_{post-pre}$ ), there may be three possible outcomes, each with a specific probability. That is, the general equation for the fixed delay change example of stochastic delay plasticity may be given by:

$$\delta(\Delta t) = \begin{cases} -D_s & \text{with probability } p_d(\Delta t) \\ D_s & \text{with probability } p_i(\Delta t) \\ 0_s & \text{with probability } 1 - [p_i(\Delta t) + p_d(\Delta t)] \end{cases} \quad (15)$$

In one aspect, stochastic delay plasticity may be based on the functional form of the delay plasticity curves  $p_d(\Delta t)$  and  $p_i(\Delta t)$ , where  $i$ =increasing and  $d$ =decreasing.  $D$  is the fixed amount of delay change, which in this aspect, may be selected to be 1. Although the delay has been selected to be 1, this is merely exemplary and not limiting. Notably, the delay may be locally updated on a time step basis, as opposed to being globally applied thereby enabling more efficient updates and improved network performance.

## 13

In some aspects, the stochastic delay plasticity may be based on piece-wise linear curves. For example, in one aspect, the equations of the piece-wise linear probability curves may be expressed as follows:

$$p_i(\Delta t) = \begin{cases} m_{i+}\Delta t + b_{i+}, & \Delta t \geq \Delta t_{stable} \\ m_{i-}\Delta t + b_{i-}, & \Delta t \leq \Delta t_{stable} \end{cases} \quad (16)$$

$$p_d(\Delta t) = \begin{cases} -m_{d+}\Delta t + b_{d+}, & \Delta t \geq \Delta t_{stable} \\ -m_{d-}\Delta t + b_{d-}, & \Delta t \leq \Delta t_{stable} \end{cases} \quad (17)$$

where  $m_{i+}$ ,  $m_{i-}$ ,  $m_{d+}$  and  $m_{d-}$  are the slopes of the piece wise linear curves,  $b_{i+}$ ,  $b_{i-}$ ,  $b_{d+}$  and  $b_{d-}$  are the curve intercepts, and  $\Delta t$  is the difference between the presynaptic spike and the postsynaptic spike.

For each curve, there are two slopes ( $m_{i+}$  and  $m_{i-}$  or  $m_{d+}$  and  $m_{d-}$ ) and two intercepts ( $b_{i+}$  and  $b_{i-}$  or  $b_{d+}$  and  $b_{d-}$ ), which are separated at a point  $\Delta t_{stable}$ . In some aspects,  $\Delta t_{stable}$  is a point at which the slopes and intercepts change.  $\Delta t_{stable}$  may define an optimally stable post-pre spike time difference. Because these curves represent probabilities, their ranges may be fixed between 0 and 1 and the domain may be set by the expected length of the pattern (e.g., 50 ms).

In terms of stochastic delay plasticity design, a set of parameters may be adjusted to control aspects of the operation of the neural network. For example, in some aspects, the parameters may include:

- (1) The sign of the slopes, which may define model behavior,
- (2) The relative magnitude of the slopes, which may control convergence,
- (3) The intercepts, which may control volatility, and
- (4) The stable point ( $\Delta t_{stable}$ ) and stable region, which may control noise performance.

Of course, this list is not exhaustive and other parameters may also be considered. Further, while these considerations/parameters may be inter-related, they may also be unrelated, and accordingly, the effect of each parameter may be related to the aspect that is to be controlled.

#### Signs of the Slopes

In one aspect, the slopes of  $p_d(\Delta t)$  (e.g.,  $m_{d+}$  and  $m_{d-}$ ) may both be negative and the slopes of  $p_i(\Delta t)$  (e.g.,  $m_{i+}$  and  $m_{i-}$ ) may both be positive. The reason for this can be seen by considering each segment of the curve separately. For example, with the  $p_i(\Delta t)$  curve when  $\Delta t \geq \Delta t_{stable}$ , the postsynaptic spike may occur after the presynaptic spike and it may be desirable to move the pre spike closer to the post spike. That is, it may be desirable to have a higher probability of increasing the synaptic delay than decreasing the synaptic delay. Furthermore, as the  $\Delta t$  increases, there may be more confidence in the desirability of increasing the delay and so the probability of increasing the delay may be increased. This may result in a positive slope ( $m_{i+}$ ). Similarly, the  $p_d(\Delta t)$  curve may have a negative slope ( $m_{d-}$ ) when  $\Delta t \leq \Delta t_{stable}$ .

For the  $p_d(\Delta t)$  curve, when  $\Delta t \geq \Delta t_{stable}$ , the postsynaptic spike may occur after the presynaptic spike, and as the time difference increases, confidence regarding desirability of increasing the delay may also increase. In some aspects, this may be equivalent to being more confident of not decreasing the delay, and so it may be desirable to lower the probability of decreasing the delay, resulting in a negative slope ( $m_{d+}$ ) for  $p_d(\Delta t)$ . Similarly, the  $p_i(\Delta t)$  curve may have a negative slope ( $m_{i-}$ ) when  $\Delta t \leq \Delta t_{stable}$ .

## 14

#### Relative Magnitudes of the Slopes

In some aspects, the values of the slopes may indicate how likely it is that the delay will change as the absolute value of  $(\Delta t - \Delta t_{stable})$  increases, and hence the rate of convergence. For example, consider the case of a presynaptic spike that occurs 40 ms before the postsynaptic spike. Initially, the synapse between this pair may have a high probability of receiving a delay increase, moving the timing of the presynaptic spike closer to the timing of the postsynaptic spike. This may tend to move the synapse's delay towards the appropriate value at a faster rate than synapses whose pre-post pairs are more closely aligned. By bringing the farthest presynaptic spikes toward the postsynaptic spike faster than the closest presynaptic spikes, they may all reach a similar value faster than if each presynaptic spike was moved at the same rate.

#### Intercept Magnitudes

In some aspects, the intercept magnitudes may define an overall probability of changing the delay. For constant slopes (e.g.,  $m_{i+}$ ,  $m_{i-}$ ,  $m_{d+}$  and  $m_{d-}$ ), as the intercept (e.g.,  $b_{i+}$ ,  $b_{i-}$ ,  $b_{d+}$  and  $b_{d-}$ ) increases, the quantity  $p_i(\Delta t) + p_d(\Delta t)$  may increase. In turn, the delays may be more likely to change on average, resulting in increased volatility. If the model is more likely to change the delay, then this may result in faster, but less reliable convergence, or may prevent convergence by causing highly variable delays. For example, if the intercepts were set so high that  $p_i(\Delta t) + p_d(\Delta t) > p_0(\Delta t)$  for all  $\Delta t$ , then the likelihood of changing the delay may be higher than the likelihood of keeping the delay the same and the convergence may be poor. Accordingly, it may be desirable to generate the intercepts in the current temporal learning model such that  $p_i(\Delta t) + p_d(\Delta t) < p_0(\Delta t)$  for all  $\Delta t$ .

Additionally, in this example, there are four intercepts in the current model specification and their values could potentially be different. In general, it may be desirable to have  $b_{d+} > b_{i+}$  and  $b_{d-} > b_{i-}$  to ensure that the probability of decreasing the delay may be higher to the left of  $\Delta t_{stable}$ . If  $b_{i+} = b_{i-}$  and  $b_{d+} = b_{d-}$ , then  $p_0(\Delta t)$  may form a triangular function with no sharp discontinuities. However, if  $b_{i+} \neq b_{i-}$  or  $b_{d+} \neq b_{d-}$ , then there may be a sharp discontinuity at  $\Delta t_{stable}$ . One reason to induce such a discontinuity may be to have a difference in volatility between the regions to the left and right of  $\Delta t_{stable}$ . One reason to have this change in volatility may be to have the stable region (to the left of  $\Delta t_{stable}$ ) be less volatile than the region to the right, which could potentially speed up convergence.

#### Stable Point and Region

The point at which the slopes and intercepts change is defined to be the most stable post-pre time difference,  $\Delta t_{stable}$ . In some aspects,  $\Delta t_{stable}$  may be defined as the point at which  $p_i(\Delta t) + p_d(\Delta t)$  is smallest. As such, the probability of not changing the synaptic delay may be the greatest. Because the probability of not changing the delay is not equal to one at this point, the synaptic delay may oscillate around the final delay value.

Furthermore, the intersection point of  $p_i(\Delta t)$  and  $p_d(\Delta t)$  may define a point at which the probability of decreasing the delay is always greater than the probability of increasing the delay. Because the delay can never reduce below one, the region to the left of the intersection point is the stable region for the model. When designing the curves, one should choose the intersection point based on the expected noise in the pattern. For example, if the intersection point is set at 1 ms, then each pre spike would occur exactly one time-step before the post spike to maintain stability. Otherwise, the  $\Delta t$  may fall to the right of the intersection point (see e.g., 908 of FIG. 9) and the delays will increase by one, which can force the post spike to occur later if the weights of those synapses are high.

15

If there is no noise in the pre spike times from trial to trial,  $\Delta t=1$  may be an acceptable intersection point. However, if it is known that the pre-spike times have a likely noise of  $\pm 3$  ms, then  $\Delta t_{stable}$  should be set to 1 and the intersection point set  $\Delta t$  greater than 3 to ensure that once the delays are reduced/minimized, increasing the delay will remain unlikely and the delays will remain stable.

In summary, while the slopes, intercepts and stable point are all related, they each have a primary function and the design of the parameter values can be constrained by this primary function.

FIG. 5 illustrates an example implementation 500 of the aforementioned operating a neural network using a general-purpose processor 502 in accordance with certain aspects of the present disclosure. Variables (neural signals), synaptic weights, system parameters associated with a computational network (neural network), delays, and frequency bin information may be stored in a memory block 504, while instructions executed at the general-purpose processor 502 may be loaded from a program memory 506. In an aspect of the present disclosure, the instructions loaded into the general-purpose processor 502 may comprise code for monitoring a timing of a presynaptic spike, monitoring a timing of a postsynaptic spike, determining a time difference between the postsynaptic spike and the presynaptic spike, and/or calculating a stochastic update of a synapse's delay based on the time difference.

FIG. 6 illustrates an example implementation 600 of the aforementioned operating a neural network where a memory 602 can be interfaced via an interconnection network 604 with individual (distributed) processing units (neural processors) 606 of a computational network (neural network) in accordance with certain aspects of the present disclosure. Variables (neural signals), synaptic weights, system parameters associated with the computational network (neural network) delays, and frequency bin information may be stored in the memory 602, and may be loaded from the memory 602 via connection(s) of the interconnection network 604 into each processing unit (neural processor) 606. In an aspect of the present disclosure, the processing unit 606 may be configured to monitor a timing of a presynaptic spike, monitor a timing of a postsynaptic spike, determine a time difference between the postsynaptic spike and the presynaptic spike, and/or calculate a stochastic update of a synapse's delay based on the time difference.

FIG. 7 illustrates an example implementation 700 of the aforementioned operating a neural network. As illustrated in FIG. 7, one memory bank 702 may be directly interfaced with one processing unit 704 of a computational network (neural network). Each memory bank 702 may store variables (neural signals), synaptic weights, and/or system parameters associated with a corresponding processing unit (neural processor) 704 delays, and frequency bin information. In an aspect of the present disclosure, the processing unit 704 may be configured to monitor a timing of a presynaptic spike, monitor a timing of a postsynaptic spike, determine a time difference between the postsynaptic spike and the presynaptic spike, and/or calculate a stochastic update of a synapse's delay based on the time difference.

FIG. 8 illustrates an example implementation of a neural network 800 in accordance with certain aspects of the present disclosure. As illustrated in FIG. 8, the neural network 800 may have multiple local processing units 802 that may perform various operations of methods described herein. Each local processing unit 802 may comprise a local state memory 804 and a local parameter memory 806 that store parameters of the neural network. In addition, the local processing unit

16

802 may have a local (neuron) model program (LMP) memory 808 for storing a local model program, a local learning program (LLP) memory 810 for storing a local learning program, and a local connection memory 812. Furthermore, as illustrated in FIG. 8, each local processing unit 802 may be interfaced with a configuration processing unit 814 for providing configurations for local memories of the local processing unit, and with a routing connection processing unit 816 that provide routing between the local processing units 802.

The neuron model includes a means for monitoring a timing of a presynaptic spike, means for monitoring a postsynaptic spike, determining means and calculating means. In one aspect, the means for monitoring a timing of a presynaptic spike, means for monitoring a postsynaptic spike, determining means and/or calculating means may be the general-purpose processor 502, program memory 506, memory block 504, memory 602, interconnection network 604, processing units 606, processing unit 704, local processing units 802, and or the routing connection processing units 816 configured to perform the functions recited. In another configuration, the aforementioned means may be any module or any apparatus configured to perform the functions recited by the aforementioned means.

According to certain aspects of the present disclosure, each local processing unit 802 may be configured to determine parameters of the neural network based upon desired one or more functional features of the neural network, and develop the one or more functional features towards the desired functional features as the determined parameters are further adapted, tuned and updated.

FIGS. 9A-9B show diagrams 900 and 950 illustrating exemplary stochastic delay plasticity curves in accordance with aspects of the present disclosure. Referring to FIG. 9A, a first curve 902 indicates a probability of increasing the delay by one (e.g.,  $p_i(\Delta t)$ ). A second curve 904 indicates a probability of decreasing the delay by one (e.g.,  $p_d(\Delta t)$ ). The first curve 902 and the second curve 904 may respectively be given by Equations 16 and 17 described above. Although, these curves are in the form of piecewise linear curves, this is merely exemplary, and any form of curve such as a higher order polynomial or a piecewise constant, for example, may alternatively be used. Furthermore, a lookup table may also be used in place of the piecewise linear curves.

A stable point ( $\Delta t_{stable}$ ) 906 is also shown. The stable point 906 corresponds to the most stable post-pre time difference. The stable point  $\Delta t_{stable}$  may be defined as the point at which the delay is least likely to be changed (i.e., point at which the probability of not changing is the highest). For example, as shown, in FIG. 9A,  $\Delta t_{stable}$  is the point at which  $p_i(\Delta t)+p_d(\Delta t)$  is smallest and hence the probability of not changing the synaptic delay is greatest.

An intersection point 908 is shown as the intersection of the stochastic plasticity curves 902 and 904. For ease of illustration, a scaled version (910) of diagram 900 is included to more clearly show the stable point  $\Delta t_{stable}$  and the intersection point 908. The intersection point 908 may define a point at which a probability of decreasing the delay is always greater than a probability of increasing the delay. In the example of FIG. 9A, because the delay may not reduce below one, the region to the left of the intersection point 908 may define a stable region for the model.

In some aspects, it may be desirable to select the intersection point based on an expected noise in the pattern. For example, if the intersection point were set at 1 ms, then each pre spike would occur exactly one time-step before the post spike to maintain stability. Otherwise, the  $\Delta t$  may fall to the right of the intersection point and the delays may increase by

17

one, which may force the post spike to occur later if the weights of the subject synapses are high.

FIG. 9B is a diagram 950 that illustrates stochastic delay plasticity curves 952, 954, 958 and 960. Referring to FIG. 9B, curve 952 shows the probability of increasing the delay by 1, while curve 954 indicates the probability of decreasing the delay by 1. The third curve 958 indicates the probability of making no change and the probability of making a change. The fourth curve 960 indicates the probability of making any change. In this exemplary diagram,  $\Delta t_{stable}$  956 may define a minimum of the probability of making a change and the maximum probability of making no change.

FIG. 10 illustrates a method 1000 for operating a neural network. In block 1002, the neuron model monitors a timing of a presynaptic spike. In block 1004, the neuron model monitors a timing of a postsynaptic spike. In block 1006, the neuron model determines a time difference between the postsynaptic spike and the presynaptic spike. Furthermore, in block 1008, the neuron model calculates a stochastic update of a synapse's delay based on the time difference.

The various operations of methods described above may be performed by any suitable means capable of performing the corresponding functions. The means may include various hardware and/or software component(s) and/or module(s), including, but not limited to, a circuit, an application specific integrated circuit (ASIC), or processor. Generally, where there are operations illustrated in the figures, those operations may have corresponding counterpart means-plus-function components with similar numbering.

As used herein, the term "determining" encompasses a wide variety of actions. For example, "determining" may include calculating, computing, processing, deriving, investigating, looking up (e.g., looking up in a table, a database or another data structure), ascertaining and the like. Additionally, "determining" may include receiving (e.g., receiving information), accessing (e.g., accessing data in a memory) and the like. Furthermore, "determining" may include resolving, selecting, choosing, establishing and the like.

As used herein, a phrase referring to "at least one of" a list of items refers to any combination of those items, including single members. As an example, "at least one of: a, b, or c" is intended to cover: a, b, c, a-b, a-c, b-c, and a-b-c.

The various illustrative logical blocks, modules and circuits described in connection with the present disclosure may be implemented or performed with a general purpose processor, a digital signal processor (DSP), an application specific integrated circuit (ASIC), a field programmable gate array signal (FPGA) or other programmable logic device (PLD), discrete gate or transistor logic, discrete hardware components or any combination thereof designed to perform the functions described herein. A general-purpose processor may be a microprocessor, but in the alternative, the processor may be any commercially available processor, controller, microcontroller or state machine. A processor may also be implemented as a combination of computing devices, e.g., a combination of a DSP and a microprocessor, a plurality of microprocessors, one or more microprocessors in conjunction with a DSP core, or any other such configuration.

The steps of a method or algorithm described in connection with the present disclosure may be embodied directly in hardware, in a software module executed by a processor, or in a combination of the two. A software module may reside in any form of storage medium that is known in the art. Some examples of storage media that may be used include random access memory (RAM), read only memory (ROM), flash memory, erasable programmable read-only memory (EPROM), electrically erasable programmable read-only

18

memory (EEPROM), registers, a hard disk, a removable disk, a CD-ROM and so forth. A software module may comprise a single instruction, or many instructions, and may be distributed over several different code segments, among different programs, and across multiple storage media. A storage medium may be coupled to a processor such that the processor can read information from, and write information to, the storage medium. In the alternative, the storage medium may be integral to the processor.

The methods disclosed herein comprise one or more steps or actions for achieving the described method. The method steps and/or actions may be interchanged with one another without departing from the scope of the claims. In other words, unless a specific order of steps or actions is specified, the order and/or use of specific steps and/or actions may be modified without departing from the scope of the claims.

The functions described herein may be implemented in hardware, software, firmware, or any combination thereof. If implemented in hardware, an example hardware configuration may comprise a processing system in a device. The processing system may be implemented with a bus architecture. The bus may include any number of interconnecting buses and bridges depending on the specific application of the processing system and the overall design constraints. The bus may link together various circuits including a processor, machine-readable media, and a bus interface. The bus interface may be used to connect a network adapter, among other things, to the processing system via the bus. The network adapter may be used to implement signal processing functions. For certain aspects, a user interface (e.g., keypad, display, mouse, joystick, etc.) may also be connected to the bus. The bus may also link various other circuits such as timing sources, peripherals, voltage regulators, power management circuits, and the like, which are well known in the art, and therefore, will not be described any further.

The processor may be responsible for managing the bus and general processing, including the execution of software stored on the machine-readable media. The processor may be implemented with one or more general-purpose and/or special-purpose processors. Examples include microprocessors, microcontrollers, DSP processors, and other circuitry that can execute software. Software shall be construed broadly to mean instructions, data, or any combination thereof, whether referred to as software, firmware, middleware, microcode, hardware description language, or otherwise. Machine-readable media may include, by way of example, random access memory (RAM), flash memory, read only memory (ROM), programmable read-only memory (PROM), erasable programmable read-only memory (EPROM), electrically erasable programmable Read-only memory (EEPROM), registers, magnetic disks, optical disks, hard drives, or any other suitable storage medium, or any combination thereof. The machine-readable media may be embodied in a computer-program product. The computer-program product may comprise packaging materials.

In a hardware implementation, the machine-readable media may be part of the processing system separate from the processor. However, as those skilled in the art will readily appreciate, the machine-readable media, or any portion thereof, may be external to the processing system. By way of example, the machine-readable media may include a transmission line, a carrier wave modulated by data, and/or a computer product separate from the device, all which may be accessed by the processor through the bus interface. Alternatively, or in addition, the machine-readable media, or any portion thereof, may be integrated into the processor, such as the case may be with cache and/or general register files.

Although the various components discussed may be described as having a specific location, such as a local component, they may also be configured in various ways, such as certain components being configured as part of a distributed computing system.

The processing system may be configured as a general-purpose processing system with one or more microprocessors providing the processor functionality and external memory providing at least a portion of the machine-readable media, all linked together with other supporting circuitry through an external bus architecture. Alternatively, the processing system may comprise one or more neuromorphic processors for implementing the neuron models and models of neural systems described herein. As another alternative, the processing system may be implemented with an application specific integrated circuit (ASIC) with the processor, the bus interface, the user interface, supporting circuitry, and at least a portion of the machine-readable media integrated into a single chip, or with one or more field programmable gate arrays (FPGAs), programmable logic devices (PLDs), controllers, state machines, gated logic, discrete hardware components, or any other suitable circuitry, or any combination of circuits that can perform the various functionality described throughout this disclosure. Those skilled in the art will recognize how best to implement the described functionality for the processing system depending on the particular application and the overall design constraints imposed on the overall system.

The machine-readable media may comprise a number of software modules. The software modules include instructions that, when executed by the processor, cause the processing system to perform various functions. The software modules may include a transmission module and a receiving module. Each software module may reside in a single storage device or be distributed across multiple storage devices. By way of example, a software module may be loaded into RAM from a hard drive when a triggering event occurs. During execution of the software module, the processor may load some of the instructions into cache to increase access speed. One or more cache lines may then be loaded into a general register file for execution by the processor. When referring to the functionality of a software module below, it will be understood that such functionality is implemented by the processor when executing instructions from that software module.

If implemented in software, the functions may be stored or transmitted over as one or more instructions or code on a computer-readable medium. Computer-readable media include both computer storage media and communication media including any medium that facilitates transfer of a computer program from one place to another. A storage medium may be any available medium that can be accessed by a computer. By way of example, and not limitation, such computer-readable media can comprise RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage or other magnetic storage devices, or any other medium that can be used to carry or store desired program code in the form of instructions or data structures and that can be accessed by a computer. In addition, any connection is properly termed a computer-readable medium. For example, if the software is transmitted from a website, server, or other remote source using a coaxial cable, fiber optic cable, twisted pair, digital subscriber line (DSL), or wireless technologies such as infrared (IR), radio, and microwave, then the coaxial cable, fiber optic cable, twisted pair, DSL, or wireless technologies such as infrared, radio, and microwave are included in the definition of medium. Disk and disc, as used herein, include compact disc (CD), laser disc, optical disc, digital versatile disc (DVD), floppy disk, and Blu-Ray® disc where disks usually reproduce data magnetically, while discs reproduce data optically with lasers. Thus, in some aspects com-

puter-readable media may comprise non-transitory computer-readable media (e.g., tangible media). In addition, for other aspects computer-readable media may comprise transitory computer-readable media (e.g., a signal). Combinations of the above should also be included within the scope of computer-readable media.

Thus, certain aspects may comprise a computer program product for performing the operations presented herein. For example, such a computer program product may comprise a computer-readable medium having instructions stored (and/or encoded) thereon, the instructions being executable by one or more processors to perform the operations described herein. For certain aspects, the computer program product may include packaging material.

Further, it should be appreciated that modules and/or other appropriate means for performing the methods and techniques described herein can be downloaded and/or otherwise obtained by a user terminal and/or base station as applicable. For example, such a device can be coupled to a server to facilitate the transfer of means for performing the methods described herein. Alternatively, various methods described herein can be provided via storage means (e.g., RAM, ROM, a physical storage medium such as a compact disc (CD) or floppy disk, etc.), such that a user terminal and/or base station can obtain the various methods upon coupling or providing the storage means to the device. Moreover, any other suitable technique for providing the methods and techniques described herein to a device can be utilized.

It is to be understood that the claims are not limited to the precise configuration and components illustrated above. Various modifications, changes and variations may be made in the arrangement, operation and details of the methods and apparatus described above without departing from the scope of the claims.

What is claimed is:

1. A method of operating a spiking neural network having a plurality of neurons coupled together with at least one synapse, comprising:

monitoring a timing of a presynaptic spike;  
monitoring a timing of a postsynaptic spike;  
determining a time difference between the postsynaptic spike and the presynaptic spike; and  
calculating a stochastic update of a delay for the at least one synapse based at least in part on the time difference.

2. The method of claim 1, in which the stochastic update is based at least in part on an evaluation of a probability function.

3. The method of claim 2, in which the probability function is based at least in part on an increase in the delay.

4. The method of claim 2, in which the probability function is based at least in part on a decrease in the delay.

5. The method of claim 2, in which at least one region of a probability distribution is parameterized.

6. The method of claim 2, in which the probability function is piecewise linear.

7. The method of claim 1, in which the update is based at least in part on a look up table.

8. The method of claim 1, in which the update is based at least in part on a calculation.

9. An apparatus for operating a spiking neural network having a plurality of neurons coupled together with at least one synapse, comprising:

a memory; and  
at least one processor coupled to the memory, the at least one processor being configured:  
to monitor a timing of a presynaptic spike;  
to monitor a timing of a postsynaptic spike;  
to determine a time difference between the postsynaptic spike and the presynaptic spike; and

## 21

to calculate a stochastic update of a delay for the at least one synapse based at least in part on the time difference.

10. The apparatus of claim 9, in which the at least one processor is configured to calculate the stochastic update based at least in part on an evaluation of a probability function.

11. The apparatus of claim 10, in which the probability function is based at least in part on an increase in the delay.

12. The apparatus of claim 10, in which the probability function is based at least in part on a decrease in the delay.

13. The apparatus of claim 10, in which at least one region of a probability distribution is parameterized.

14. The apparatus of claim 10, in which the probability function is piecewise linear.

15. The apparatus of claim 9, in which the at least one processor is configured to calculate the stochastic update based at least in part on a look up table.

16. The apparatus of claim 9, in which the at least one processor is configured to calculate the stochastic update based at least in part on a calculation.

17. An apparatus for operating a spiking neural network having a plurality of neurons coupled together with at least one synapse, comprising:

means for monitoring a timing of a presynaptic spike;

means for monitoring a timing of a postsynaptic spike;

means for determining a time difference between the postsynaptic spike and the presynaptic spike; and

means for calculating a stochastic update of a delay for the at least one synapse based at least in part on the time difference.

## 22

18. The apparatus of claim 17, in which the means for calculating the stochastic update calculates the stochastic update based at least in part on an evaluation of a probability function.

19. The apparatus of claim 18, in which the probability function is based at least in part on an increase in the delay.

20. The apparatus of claim 18, in which the probability function is based at least in part on a decrease in the delay.

21. The apparatus of claim 18, in which at least one region of a probability distribution is parameterized.

22. The apparatus of claim 18, in which the probability function is piecewise linear.

23. The apparatus of claim 17, in which the means for calculating the stochastic update calculates the stochastic update based at least in part on a look up table.

24. The apparatus of claim 17, in which the means for calculating the stochastic update calculates the stochastic update based at least in part on a calculation.

25. A computer program product, comprising:  
a non-transitory computer readable medium have encoded thereon program code, the program code comprising:  
program code to monitor a timing of a presynaptic spike;  
program code to monitor a timing of a postsynaptic spike;  
program code to determine a time difference between the postsynaptic spike and the presynaptic spike; and  
program code to calculate a stochastic update of a delay for at least one synapse based at least in part on the time difference.

\* \* \* \* \*